REMARKS

Claims 1-11 and 16-18 will be pending in the current Application upon entering this Amendment. Claims 12-15 have been cancelled, and claims 20-28 have been added. All the amendments herein have been made in order to clarify the claims and not for prior art reasons. Applicants also submit that (1) no amendment made was related to the statutory requirements of patentability unless expressly stated herein, and (2) no amendment made was for the purpose of narrowing the scope of any claim, unless Applicants have argued herein that such amendment was made to distinguish over a particular reference or combination of references.

Rejection of claims 14-19 under 35 U.S.C. 101

Applicants have cancelled claims 14-15 and submit that claims 16-19 are directed to statutory subject matter and should therefore be patentable under 35 U.S.C. 101. With respect to claims 16-19, Applicants submit that a processor instruction is not merely a data structure per se, as stated by the Examiner, but instead performs a practical application. For example, as known in the art, processor instructions are used in writing code to perform a variety of different functions (where the processor instructions use their corresponding fields to perform their corresponding functions), and is not simply an arrangement of data (such as a data structure). That is, a processor instruction is *not* a data structure. A claim to a processor instruction is not merely the manipulation of abstract ideas, but instead has practical applications in the area of processor execution. Therefore, Applicants submit that claims 16-19 are directed to statutory subject matter.

Rejection of claims 1-11 under 35 U.S.C. 103

Applicants respectfully submit that claims 1-11 are patentable under 35 U.S.C. 103 over Stoodley, "Software Pipelining Loops with Conditional Branches."

With respect to claim 1, Applicants submit that claim 1 is patentable over Stoodley since Stoodley does not teach or suggest each and every element of claim 1. For example, claim 1 includes "an instruction fetching mechanism that retrieves the set of instructions" where at least

one of the set of instructions comprises "a single instruction that provides for execution of other instructions of the set of instructions in accordance with multiple looping constructs." The Examiner states that Stoodley addresses a single looping construct but lacks teaching "multiple looping constructs." However, the Examiner proceeds to state that "Stoodley suggests looping of more than one condition branch resulting in more than two iteration paths (loops)." However, the existence of multiple iteration paths does not teach or suggest multiple loop constructs. That is, it is irrelevant how many iteration paths may exist within a same loop construct. For example, the C code in Figure 1 of Stoodley illustrates a single looping construct (the "for" loop) having an if-then-else statement in the body of the single looping construct. The true and false paths of that if-then-else statement may result in different iteration paths, but this does not suggest the use of a *single instruction* that provides for execution in accordance with *multiple looping constructs*. (Note also that the operations in the C code, such as the "for" operation, are not processor instructions that are executed by an execution unit of a processor. That is, they must be compiled or otherwise translated into processor instructions such as those illustrated beneath the C code in Figure 1 of Stoodley in order to execute on a processor.) Even if one modifies an instruction of Stoodley to perform multiple iteration paths, as suggested by the Examiner, the resulting multiple iteration paths do not result in multiple looping constructs. Therefore, there is no teaching or suggestion in Stoodley of a single instruction corresponding to multiple looping constructs nor would one of ordinary skill in the art be motivated to modify Stoodley to obtain such a single instruction because the issue of different iteration paths (e.g. true/false paths) is not relevant to multiple looping constructs. Therefore, for at least these reasons, Applicants submit that claim 1 is patentable over Stoodley.

With respect to claim 2, the Examiner takes official notice that "initializes a plurality of loops for later execution" is a well known feature in the art and provides the example of exception handling. However, Applicants respectfully disagree. Firstly, exception handling does not necessarily initialize a plurality of loops for later execution. Furthermore, exception handling does not teach or suggest a single instruction (such as a processor instruction fetched and executed by a processor, as claimed in claim 1) which initializes a plurality of loops for later execution. Therefore, as claimed in claim 2, Applicants submit that a single instruction that "initializes a plurality of loops for later execution" is not well known in the art. If the Examiner disagrees, Applicants request that the Examiner provide a reference to support his position.

Applicants submit that claims 2-5 which depend directly or indirectly from allowable claim 1 are also allowable for at least those reasons provided with respect to claim 1.

With respect to claim 6, Applicants submit that claim 6 is patentable over Stoodley since Stoodley does not teach or suggest each and every element of claim 6. For example, claim 6 claims "initializing a plurality of dedicated loop storage elements corresponding to a plurality of different loops to be executed using a single instruction." Stoodley does not teach or suggest the use of a single processor instruction capable of initializing a plurality of different loops. For example, referring to the processor instructions illustrated beneath the C code in Figure 1 in Stoodley, none of the processor instructions is capable of initializing a *plurality of dedicated loop storage elements* corresponding to a *plurality of different loops* to be executed using a single instruction. (Note that the operations in the C code are not processor instructions. That is, they must be compiled or otherwise translated into processor instructions such as those illustrated beneath the C code in Figure 1 of Stoodley in order to execute on a processor. Also note that there is no teaching or suggestion in Stoodley of loop storage elements as claimed in claim 6.) Also, as discussed above, the existence of multiple iteration paths for a given loop is not relevant to the use of a single instruction for executing a plurality of loops because the multiple iteration paths discussed in Stoodley still correspond to a same loop (i.e. to a single loop, and not multiple loops). Therefore, for at least these reasons, Applicants submit that claim 6 is patentable over Stoodley.

Applicants submit that claims 7 and 8 which depend directly or indirectly from allowable claim 6 are also allowable for at least those reasons provided with respect to claim 6.

With respect to claim 9, Applicants submit that claim 9 is patentable over Stoodley since Stoodley does not teach or suggest each and every element of claim 9. Note that claim 9 includes elements that differ from claim 1; therefore, although some of the arguments provided above with respect to claim 1 apply to claim 9, claim 9 also includes different claim elements. For example, claim 9 claims "determining a loop type for a single instruction that is to execute a plurality of different loops, the loop type comprising one of a conditional and non-conditional type of loop termination." Firstly, as described above, Stoodley does not teach or suggest the single instruction that is to execute a plurality of different loops as claimed in claim 9. Secondly, there is no teaching or suggestion in Stoodley of determining a loop type, as claimed in claim 9. The Examiner (in the section discussing claim 10) agrees that Stoodley lacks teaching "from one

of a conditional and nonconditional type of loop termination for a loop." However, the Examiner states that Stoodley "suggests looping of more than one condition branch resulting in more than two iteration paths." However, Applicants respectfully submit that the teaching of multiple iteration paths does not suggest the use of a loop type as claimed in claim 9. That is, the true and false paths of the if-then-else statement in the C code of Stoodley may result in different iteration paths of the same loop, but the type of instructions (conditional or not) within the loop does not teach or suggest determining a conditional or nonconditional loop type, as claimed in claim 9.

With respect to claims 10 and 11, the same arguments provided above with respect to claim 9 also apply to claims 10 and 11, since each of claims 10 and 11 include, for example, "determining a loop type" as described above in reference to claim 9.

Rejection of claims 16-19 under 35 U.S.C. 103

Applicants respectfully submit that claims 16-19 are patentable under 35 U.S.C. 103 over Albert, "Data Parallel Computers in the FORALL Statements." Claim 16 claims a processor instruction having a first field that indicates a first termination condition for a first execution loop and a second field that indicates a second termination condition for a second execution loop. The Examiner states that Albert's "forall" operation teaches this limitation. However, Applicants respectfully disagree. The Examiner does state that Albert does not address the data structure in a manner of a processor instruction but that it would be "an intended use of Albert's data structure because multiple iteration paths are often dealt in any computer-programming algorithm." However, the concept of multiple iteration paths is again irrelevant to the processor instruction of claim 16. Furthermore, as discussed above, claim 16 is not claiming a data structure but a processor instruction. Although Albert illustrates a high level FORTRAN operation "forall", this operation is not a processor instruction. That is, for this "for all" operation to execute on a processor, it must be compiled or otherwise translated into processor instructions, and, as a result, each loop termination (e.g. I and J) would not be defined as fields in a same instruction, but would require different processor instructions. Therefore, Albert does not teach or suggest a processor instruction having multiple fields corresponding to multiple termination conditions for multiple loops. Therefore, for at least these reasons, Applicants submit that claim 16 is patentable over Albert.

Applicants submit that claims 17-19 which depend directly or indirectly from allowable claim 16 are also allowable for at least those reasons provided with respect to claim 16.

Added Claims 20-28

With respect to added claim 20, none of the cited references teach a single instruction as claimed in claim 20. Note that the single instruction is a processor instruction (i.e. it is fetched and decoded by a processor) and thus is not simply a high level operation. That is, the "for" statement in Stoodley or the Fortran "forall" operation in Albert are not processor instructions. Therefore, in the cited references, multiple processor instructions, rather than a single instruction, would have to be fetched and decoded in order to perform those operations. Furthermore, as discussed above, none of the cited references teach or suggest "in response to decoding the single instruction, using information provided by the single instruction to initialize a plurality of loop storage elements corresponding to the first loop and the second loop." Therefore, for at least these reasons, none of the cited references teach or suggest a single instruction which initializes loop storage elements as claimed in claim 20.

Claims 21-28 depend directly or indirectly from allowable claim 20 and are therefore also allowable for at least those reasons provided with respect to claim 20.

Conclusion

Although Applicants may disagree with statements made by the Examiner in reference to the claims and the cited references, Applicants are not discussing all these statements in the current Office Action, yet reserve the right to address them at a later time if necessary.

Applicants respectfully solicit allowance of the pending claims. Contact me if there are any issues regarding this communication or the current Application.

Respectfully submitted,

By: _____

Joanna G. Chiu
Attorney of Record
Reg. No.: 43,629
Telephone: (512) 996-6839
Fax No.: (512) 996-6854
Email: Joanna.Chiu@Motorola.com

SEND CORRESPONDENCE TO:

Motorola, Inc.
Law Department

Customer Number: 23125